

**Amendments to the Claims**

This listing of claims will replace all prior versions and listings of claims in the above-identified application.

**Listing of Claims**

1. **(Currently Amended)** A method for finding a path in a network, wherein said network comprises a plurality of nodes and a plurality of links and each one of said plurality of nodes is coupled to at least one other of said plurality of nodes by at least one of said plurality of links, comprising:

generating at least one path cost data set, said path cost data set representing a path cost between a root node of said nodes and destination node of said nodes, wherein said path begins at said root node and ends at said destination node; and

accessing said at least one path cost data set wherein

said generating and said accessing are performed in such a manner that a minimum-hop path and a minimum-cost path can be determined from said at least one path cost data set,

said minimum-hop path represents a path between said root node and said destination node having a minimum number of hops, and

said minimum-cost path represents a path between said root node and said destination node having a minimum cost, and

said minimum cost comprises a cost of a bandwidth available along at least a portion of said path.

2. **(Original)** The method of claim 1, further comprising:  
storing said at least one path cost data set in a path storage area such that said at least one path cost data set can be accessed to determine said minimum-hop path and said minimum-cost path.
3. **(Original)** The method of claim 2, further comprising:  
allocating said path storage area in a data structure that facilitates said access to determine said minimum-hop path and said minimum-cost path.

4. **(Original)** The method of claim 1, further comprising:  
storing said at least one path cost data set in a data structure, wherein  
said data structure is a two-dimensional array of entries arranged in a plurality  
of rows and a plurality of columns,  
each one of said rows in said data structure corresponds to one of said  
plurality of nodes, and  
each one of said columns in said data structure corresponds to a given hop  
count.
5. **(Original)** The method of claim 4, further comprising:  
determining said minimum-hop path to said destination node by:  
traversing a one of said rows corresponding to said destination node from a  
first column of said columns to a second column of said columns, and  
storing path information representing said minimum-hop path while traversing  
said data structure from said second column to said first column, said  
second column being a first one of said columns encountered when  
traversing said one of said rows from said first column to said second  
column having non-default cost entry.
6. **(Original)** The method of claim 5, wherein said first column corresponds to said root  
node.
7. **(Original)** The method of claim 4, further comprising:  
determining said minimum-cost path to said destination node by:  
identifying a minimum-cost column of said columns, said minimum-cost  
column having a lowest cost entry of all of said columns in a one of  
said rows corresponding to said destination node, and  
storing path information representing said minimum-cost path while  
traversing said data structure from said minimum-cost column to a first  
column of said columns.
8. **(Original)** The method of claim 7, wherein said first column corresponds to said root  
node.

9. (Canceled)

10. (Previously Presented) A method of finding a path in a network comprising:  
creating a path table, wherein:

said path table comprises a first number of rows and a second number of  
columns,

said network comprises a plurality of nodes and a plurality of links,  
each one of said plurality of nodes is coupled to at least one other of said  
plurality of nodes by at least one of said plurality of links, and  
said path begins at a root node of said plurality of nodes;

processing each row in a first column of said second number of columns, wherein  
said processing said each row in said first column of said second number of  
columns comprises:

for said each row in said first column of said second number of columns,  
wherein a selected node of said plurality of nodes corresponds to said  
row in said first column,  
if said selected node is a neighbor of said root node,

storing a first link cost in a first cost entry, wherein  
said first link cost is a link cost of a first one of said plurality of  
links,

said first one of said plurality of links is between said root node  
and said selected node, and

said first cost entry is a cost entry of said row in said first  
column,

storing a root node identifier in a first previous node  
entry, wherein said root node identifier  
represents said root node and said first previous  
node entry is a previous node entry of said row  
in said first column, and

storing a node identifier representing said selected node  
in a storage area, else,

storing a maximum cost value in said first cost entry,  
and

storing a null value in said first previous node entry  
 said processing each row in said first column of said second number of columns  
 resulting in said first column containing corresponding first connectivity  
 information; and  
 processing each remaining column, wherein  
 said each remaining column is a one of said second number of columns other  
 than said first column, and  
 said processing each remaining column resulting in said first column  
 containing corresponding subsequent connectivity information.

11. **(Original)** The method of claim 10, wherein said processing said each remaining column comprises:

for said each remaining column,  
 if said storage area is not empty,  
 copying each row of a preceding column to a corresponding row of  
 said remaining column, said preceding column being a one of  
 said second number of columns other than said remaining  
 column,  
 for each stored node identifier, said stored node identifier being stored  
 in said storage area and corresponding to a current node of said  
 plurality of nodes,  
 removing said stored node identifier from said storage area,  
 for each neighboring node, said neighboring node being a neighbor of said  
 current node,  
 adding a neighboring link cost to a preceding path cost in order to  
 yield an alternate path cost, wherein  
 said neighboring link cost is a link cost of a second one of said  
 plurality of links, said second one of said plurality of links  
 being between said current node and said neighboring  
 node, and  
 said preceding path cost is a cost value stored in a cost entry of a  
 row of said preceding column, said row of said preceding  
 column corresponding to said current node,

if said alternate path cost is less than a cost value stored in a current cost entry,  
 storing said alternate path cost in said current cost entry, said current cost entry being a cost entry of a row of said remaining column, said row of said remaining column corresponding to said neighboring node,  
 storing a node identifier representing said current node in a previous node entry of said row of said remaining column,  
 and  
 storing a node identifier representing said neighboring node in said storage area.

12. **(Original)** The method of claim 11, further comprising:  
 identifying said root node, wherein said root node stores a topology database comprising:  
 connectivity information regarding which ones of said plurality of nodes are coupled to which other ones of said plurality of nodes, and  
 a link cost for each one of said plurality of links.
13. **(Original)** The method of claim 12, further comprising:  
 generating said topology database by:  
 for each one of said plurality of nodes,  
 identifying, at least one neighboring node that is a neighbor of said each one of said plurality of nodes and at least one neighboring link that couples said at least one neighboring node to said each one of said plurality of nodes, wherein said plurality of nodes includes said at least one neighboring node and said plurality of links includes said at least one neighboring link,  
 determining a link cost for each one of said at least one neighboring link, and  
 storing said link cost at said each one of said plurality of nodes; and  
 distributing said connectivity information for each one of said plurality of nodes to other of said plurality of nodes.

14.     **(Original)** The method of claim 13, wherein said link cost is a physical length of a corresponding one of said plurality of links.
15.     **(Original)** The method of claim 13, wherein said link cost is a bandwidth of a corresponding one of said plurality of links.
16.     **(Original)** The method of claim 11, wherein said first number is equal to a number of nodes in said plurality of nodes.
17.     **(Original)** The method of claim 11, wherein each one of said second number of columns corresponds to a number of hops.
18.     **(Original)** The method of claim 17, wherein said number of hops is equal to a maximum number of hops.
19.     **(Original)** The method of claim 18, wherein said maximum number of hops is a maximum number of hops possible when using said method.
20.     **(Original)** The method of claim 18, wherein said maximum number of hops is a maximum number of hops selected by a user.
21.     **(Original)** The method of claim 11, wherein said each neighboring node is one of said plurality of nodes that is a neighbor of said current node.
22.     **(Original)** The method of claim 11, wherein said stored node identifier is already stored in said storage area at a time when processing of said remaining column is begun.
23.     **(Original)** The method of claim 11, wherein said preceding path cost is a cost of a path between said root node and said current node.
24.     **(Original)** The method of claim 11, wherein said storage area is a queue.

25.     **(Original)** The method of claim 11, further comprising:  
selecting a destination node from said plurality of nodes, said destination node being  
a one of said plurality of nodes other than said root node; and  
determining a minimum number of hops between said root node and said destination  
node by  
setting said minimum number of hops to one, and  
for each one of said second number of columns, incrementing said minimum  
number of hops by one if a cost entry in a row of said one of said  
second number of columns is equal to said maximum cost value, said  
row of said one of said second number of columns corresponding to  
said destination node.
26.     **(Original)** The method of claim 11, further comprising:  
selecting a destination node from said plurality of nodes, said destination node being  
a one of said plurality of nodes other than said root node; and  
determining a minimum cost of said path, said path being between said root node and  
said destination node, by searching a one of said first number of rows  
corresponding to said destination node for a one of said second number of  
columns having a smallest cost entry of cost entries of said second number of  
columns in said one of said first number of rows.
27.     **(Original)** The method of claim 11, wherein each one of said first number of rows  
corresponds to a one of said plurality of nodes.
28.     **(Original)** The method of claim 11, wherein each one of said second number of  
columns corresponds to a corresponding number of hops from said root node and said second  
number of columns is arranged in a monotonically increasing order with regard to said  
number of hops.
29.     **(Original)** The method of claim 27, wherein said preceding column corresponds to a  
number of hops that is one hop less than a number of hops corresponding to said remaining  
column.

30. **(Original)** The method of claim 28, further comprising:  
selecting a destination node from said plurality of nodes, said destination node being  
a one of said plurality of nodes other than said root node; and  
determining a minimum number of hops between said root node and said destination  
node by  
counting a number of columns from said first column to a first non-maximum-  
cost column, inclusive, said first non-maximum-cost column being a  
first one of said second number of columns for which a cost entry in a  
one of said first number of rows corresponding to said destination  
node is not said maximum cost value.
31. **(Original)** The method of claim 28, further comprising:  
selecting a destination node from said plurality of nodes, said destination node being  
a one of said plurality of nodes other than said root node;  
storing a destination node identifier in said path storage area, said destination node  
identifier representing said destination node;  
setting a current node identifier to said destination node identifier; and  
for each intermediate column, proceeding from a first non-maximum-cost column to  
said first column, wherein  
said intermediate column is a one of said second number of columns  
between said first column and said first non-maximum-cost  
column, inclusive, and  
said first non-maximum-cost column is a first one of said second  
number of columns, when proceeding from said first column to  
said first non-maximum-cost column, for which a cost entry in  
a row corresponding to said destination node is not said  
maximum cost value,  
storing, in said path storage area, a previous node identifier stored in a  
previous node entry of a row of said intermediate column, wherein  
said row of said intermediate column corresponds to a one of said  
plurality of nodes represented by said current node identifier, and  
setting a current node identifier to said previous node identifier.

32. **(Original)** The method of claim 28, further comprising:  
selecting a destination node from said plurality of nodes, said destination node being  
a one of said plurality of nodes other than said root node;  
storing a destination node identifier in said path storage area, said destination node  
identifier representing said destination node;  
setting a current node identifier to said destination node identifier;  
searching a one of said first number of rows corresponding to said destination node  
for a minimum path-cost column, wherein  
said minimum path-cost column is a one of said second number of columns  
having a smallest cost entry of said second number of columns in said  
one of said first number of rows; and  
for each intermediate column, proceeding from said minimum path-cost column to  
said first column, wherein said intermediate column is a one of said second  
number of columns between said first column and said minimum path-cost  
column, inclusive,  
storing, in said path storage area, a previous node identifier stored in a  
previous node entry of a row of said intermediate column, wherein  
said row of said intermediate column corresponds to a one of said  
plurality of nodes represented by said current node identifier, and  
setting a current node identifier to said previous node identifier.
33. **(Original)** The method of claim 11, wherein said link cost is a physical length of a  
corresponding one of said each one of said at least one neighboring links.
34. **(Original)** The method of claim 11, wherein said link cost is a bandwidth a  
corresponding one of said each one of said at least one neighboring links is configured to  
carry.

35. **(Original)** The method of claim 11, wherein:
- said storing said node identifier representing said current node in said previous node entry of said row of said remaining column further comprises:
    - storing a node identifier stored in a next node entry of said row of said preceding column in a next node entry of said row of said remaining column; and
  - said method further comprises:
    - for each row in said first column,
      - if said corresponding node is a neighbor of said root node,
        - storing a node identifier representing said corresponding node in a first next node entry, wherein said first next node entry is a next node entry of said row in said first column, and
      - else,
        - storing a null value in said first next node entry.

36. **(Canceled)**

37. (Previously Presented) A method of finding a path in a network comprising:  
creating a path vector, wherein:

said path vector comprises a first number of rows,  
said network comprises a plurality of nodes and a plurality of links,  
each one of said plurality of nodes is coupled to at least one other of said  
plurality of nodes by at least one of said plurality of links, and  
said path begins at a root node of said plurality of nodes;

for a first hop of a maximum number of hops, processing each row in said first  
number of rows for said first hop, a selected node of said plurality of nodes  
corresponding to said row wherein said processing each row in said first  
number of rows for said first hop comprises:

for said first hop of said maximum number of hops,

for each row in said first number of rows, a selected node of said  
plurality of nodes corresponding to said row,

if said selected node is a neighbor of said root node,

storing a first link cost in a first cost entry, wherein

said first link cost is a link cost of a first one of said  
plurality of links,

said first one of said plurality of links is between said  
root node and said selected node, and

said first cost entry is a cost entry of said row, and

storing a node identifier representing said selected node in a  
storage area,

else

storing a maximum cost value in said first cost entry, and

for each remaining hop of said maximum number of hops, processing said each row  
in said first number of rows for said each remaining hop.

38. (Original) The method of claim 37, wherein said processing said each row in said first number of rows for said each remaining hop comprises:

for said each remaining hop of said maximum number of hops,

if said storage area is not empty,

for each stored node identifier, said stored node identifier being stored in said storage area and representing a current node of said plurality of nodes, removing said stored node identifier from said storage area,

for each neighboring node, said neighboring node being a neighbor of said current node,

adding a neighboring link cost to a preceding path cost in order to yield an alternate path cost, wherein

said neighboring link cost is a link cost of a second one of said plurality of links, said second one of said plurality of links being between said current node and said neighboring node, and

said preceding path cost is a cost value stored in a cost entry of a row of said first number of rows corresponding to said current node,

if said alternate path cost is less than a cost value stored in a current cost entry,

storing said alternate path cost in said current cost entry, said current cost entry being a cost entry of a row of said first number of rows corresponding to said neighboring node,

storing a node identifier representing said current node in a previous node entry of said row of said first number of rows corresponding to said neighboring node, and

storing a node identifier representing said neighboring node in said storage area.

39.     **(Original)** The method of claim 38, wherein said first number is equal to a number of nodes in said plurality of nodes.
40.     **(Original)** The method of claim 38, wherein said first maximum number of hops is a maximum number of hops possible in said network.
41.     **(Original)** The method of claim 38, wherein said maximum number of hops is a maximum number of hops selected by a user.
42.     **(Original)** The method of claim 38, wherein said each neighboring node is a one of said plurality of nodes that is a neighbor of said current node.
43.     **(Original)** The method of claim 38, wherein said stored node identifier is already stored in said path storage area at a time when processing of said remaining column is begun.
44.     **(Original)** The method of claim 38, wherein said preceding path cost is a cost of a path between said root node and said current node.
45.     **(Original)** The method of claim 38, wherein said storage area is a queue.

46. **(Currently Amended)** A computer system comprising:  
 a processor coupled to a network, wherein said network comprises a plurality of nodes and a plurality of links and each one of said plurality of nodes is coupled to at least one other of said plurality of nodes by at least one of said plurality of links;  
 computer readable medium coupled to said processor; and  
 computer code, encoded in said computer readable medium, configured to cause said processor to find a path in said network by virtue of being configured to cause said processor to:  
 generate at least one path cost data set, said path cost data set representing a path cost between a root node of said nodes and destination node of said nodes, wherein said path begins at said root node and ends at said destination node; and  
 access said at least one path cost data set wherein  
 said generating and said accessing are performed in such a manner that  
 a minimum-hop path and a minimum-cost path can be  
 determined from said at least one path cost data set,  
 said minimum-hop path represents a path between said root node and  
 said destination node having a minimum number of hops, and  
 said minimum-cost path represents a path between said root node and  
 said destination node having a minimum cost, and  
said minimum cost comprises a cost of a bandwidth available along at  
least a portion of said path.
47. **(Original)** The computer system of claim 46, wherein said computer code is further configured to cause said processor to:  
 store said at least one path cost data set in a path storage area such that said at least one path cost data set can be accessed to determine said minimum-hop path and said minimum-cost path.

48. **(Original)** The computer system of claim 47, wherein said computer code is further configured to cause said processor to:

allocate said path storage area in a data structure that facilitates said access to  
determine said minimum-hop path and said minimum-cost path.

49. **(Original)** The computer system of claim 46, wherein said computer code is further configured to cause said processor to:

store said at least one path cost data set in a data structure, wherein  
said data structure is a two-dimensional array of entries arranged in a plurality  
of rows and a plurality of columns,  
each one of said rows in said data structure corresponds to one of said  
plurality of nodes, and  
each one of said columns in said data structure corresponds to a given hop  
count.

50. **(Original)** The computer system of claim 49, wherein said computer code is further configured to cause said processor to:

determine said minimum-hop path to said destination node by virtue of being  
configured to cause said processor:  
traverse a one of said rows corresponding to said destination node from a first  
column of said columns to a second column of said columns, and  
store path information representing said minimum-hop path while traversing  
said data structure from said second column to said first column, said  
second column being a first one of said columns encountered when  
traversing said one of said rows from said first column to said second  
column having non-default cost entry.

51. **(Original)** The computer system of claim 50, wherein said first column corresponds to said root node.

52. **(Original)** The computer system of claim 49, wherein said computer code is further configured to cause said processor to:

determine said minimum-cost path to said destination node by virtue of being

configured to cause said processor:

identify a minimum-cost column of said columns, said minimum-cost column

having a lowest cost entry of all of said columns in a one of said rows

corresponding to said destination node, and

store path information representing said minimum-cost path while traversing

said data structure from said minimum-cost column to a first column

of said columns.

53. **(Original)** The computer system of claim 52, wherein said first column corresponds to said root node.

54. **(Currently Amended)** A computer program product for finding a path in said network, wherein said network comprises a plurality of nodes and a plurality of links and each one of said plurality of nodes is coupled to at least one other of said plurality of nodes by at least one of said plurality of links and said computer program product is encoded in computer readable media and comprising:

a first set of instructions, executable on a computer system, configured to generate at least one path cost data set, said path cost data set representing a path cost between a root node of said nodes and destination node of said nodes, wherein said path begins at said root node and ends at said destination node; and

a second set of instructions, executable on said computer system, configured to access said at least one path cost data set wherein

said generation and said access are performed in such a manner that a

minimum-hop path and a minimum-cost path can be determined from

said at least one path cost data set,

said minimum-hop path represents a path between said root node and said

destination node having a minimum number of hops, and

said minimum-cost path represents a path between said root node and said

destination node having a minimum cost, and

said minimum cost comprises a cost of a bandwidth available along at least a portion of said path.

55. **(Original)** The computer program product of claim 54, further comprising:  
a third set of instructions, executable on said computer system, configured to store  
said at least one path cost data set in a path storage area such that said at least  
one path cost data set can be accessed to determine said minimum-hop path  
and said minimum-cost path.
56. **(Original)** The computer program product of claim 55, further comprising:  
a fourth set of instructions, executable on said computer system, configured to  
allocate said path storage area in a data structure that facilitates said access to  
determine said minimum-hop path and said minimum-cost path.
57. **(Original)** The computer program product of claim 54, further comprising:  
a third set of instructions, executable on said computer system, configured to store  
said at least one path cost data set in a data structure, wherein  
said data structure is a two-dimensional array of entries arranged in a plurality  
of rows and a plurality of columns,  
each one of said rows in said data structure corresponds to one of said  
plurality of nodes, and  
each one of said columns in said data structure corresponds to a given hop  
count.
58. **(Original)** The computer program product of claim 57, further comprising:  
a fourth set of instructions, executable on said computer system, configured to  
determine said minimum-hop path to said destination node, said fourth set of  
instructions comprising:  
a first subset of instructions, executable on said computer system, configured  
to traverse a one of said rows corresponding to said destination node  
from a first column of said columns to a second column of said  
columns, and  
a second subset of instructions, executable on said computer system,  
configured to store path information representing said minimum-hop

path while traversing said data structure from said second column to said first column, said second column being a first one of said columns encountered when traversing said one of said rows from said first column to said second column having non-default cost entry.

59. **(Original)** The computer program product of claim 58, wherein said first column corresponds to said root node.

60. **(Currently Amended)** The computer program product of claim 57, further comprising:

a fourth set of instructions, executable on said computer system~~[[,]] configured to a first subset of instructions, executable on said computer system~~, configured to determine said minimum-cost path to said destination node, said fourth set of instructions comprising:

a first subset of instructions, executable on said computer system, configured to identify a minimum-cost column of said columns, said minimum-cost column having a lowest cost entry of all of said columns in a one of said rows corresponding to said destination node, and

a second subset of instructions, executable on said computer system, configured to store path information representing said minimum-cost path while traversing said data structure from said minimum-cost column to a first column of said columns.

61. **(Original)** The computer program product of claim 60, wherein said first column corresponds to said root node.

62. **(Currently Amended)** A method for finding a path in a network, wherein said network comprises a plurality of nodes and a plurality of links and each one of said plurality of nodes is coupled to at least one other of said plurality of nodes by at least one of said plurality of links, comprising:

generating at least one path cost data set, wherein,

said path cost data set stores a path cost of a path between a root node of said nodes and destination node of said nodes,

said path is in a network,

said network comprises a plurality of nodes and a plurality of links,

each one of said plurality of nodes is coupled to at least one other of said plurality of nodes by at least one of said plurality of links, and

said nodes comprise said root node and said destination node;

a structure of said path cost data set represents a hop count of a path between said root node and said destination node, and

said path begins at said root node and ends at said destination node

storing said at least one path cost data set in a data structure, wherein

a first attribute of said data structure represents a hop count of said path, and

a second attribute of said data structure represents each node of said path; and

accessing said at least one path cost data set, wherein,

said generating and said accessing are performed in such a manner that a

minimum-hop path and a minimum-cost path can be determined from said at least one path cost data set,

said minimum-hop path represents a path between said root node and said destination node having a minimum number of hops, and

said minimum-cost path represents a path between said root node and said destination node having a minimum cost.

63. **(Currently Amended)** The method of claim 62, ~~further comprising~~ wherein said storing comprises:

storing said at least one path cost data set in a path storage area such that said at least one path cost data set can be accessed to determine said minimum-hop path and said minimum-cost path.

64. **(Currently Amended)** The method of claim 63, further comprising:  
allocating said path storage area in **[[a]]** said data structure **[[that]]**, wherein said data structure facilitates said access to determine said minimum-hop path and said minimum-cost path.
65. **(Currently Amended)** The method of claim 62, ~~further comprising:~~ wherein:  
~~storing said at least one path cost data set in a data structure, wherein~~  
said data structure is a two-dimensional array of entries arranged in a plurality of rows and a plurality of columns,  
each one of said rows in said data structure corresponds to one of said plurality of nodes, and  
each one of said columns in said data structure corresponds to a given hop count.
66. **(Previously Presented)** The method of claim 65, further comprising:  
determining said minimum-hop path to said destination node by:  
traversing a one of said rows corresponding to said destination node from a first column of said columns to a second column of said columns, and  
storing path information representing said minimum-hop path while traversing said data structure from said second column to said first column, said second column being a first one of said columns encountered when traversing said one of said rows from said first column to said second column having non-default cost entry.
67. **(Previously Presented)** The method of claim 66, wherein said first column corresponds to said root node.

68. **(Previously Presented)** The method of claim 65, further comprising:  
determining said minimum-cost path to said destination node by:  
    identifying a minimum-cost column of said columns, said minimum-cost  
        column having a lowest cost entry of all of said columns in a one of  
        said rows corresponding to said destination node, and  
    storing path information representing said minimum-cost path while  
        traversing said data structure from said minimum-cost column to a first  
        column of said columns.
69. **(Previously Presented)** The method of claim 68, wherein said first column  
corresponds to said root node.